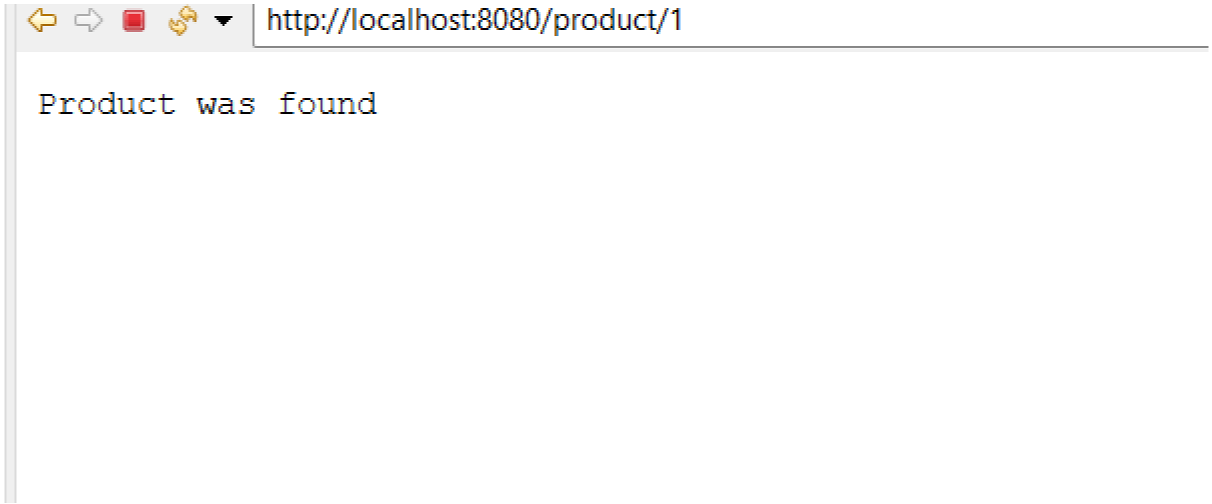
Practice Question-1

```
package com.demo.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import com.demo.exception.ProductNotFoundException;
@Controller
public class MainController {
@RequestMapping(value = "/product/{id}", method = RequestMethod.GET)
@ResponseBody
public String getProduct(@PathVariable("id") String id) {
if (id.contentEquals("0"))
throw new ProductNotFoundException();
return "Product was found";
}
}
```



http://localhost:8080/product/1

```
Product was found
```

Practice Question -2

```
package com.demo.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.client.RestTemplate;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import com.demo.exception.ProductNotFoundException;
import com.pstwo.beans.Quote;
@Controller
```

```
public class MainController {
@RequestMapping("/")
@ResponseBody
public String index() {
RestTemplate restTemplate = new RestTemplate();
Quote quote =
restTemplate.getForObject("https://gturnquist-quoters.cfapps.io/api/random",
Quote.class);
return quote.toString();
}

}
```

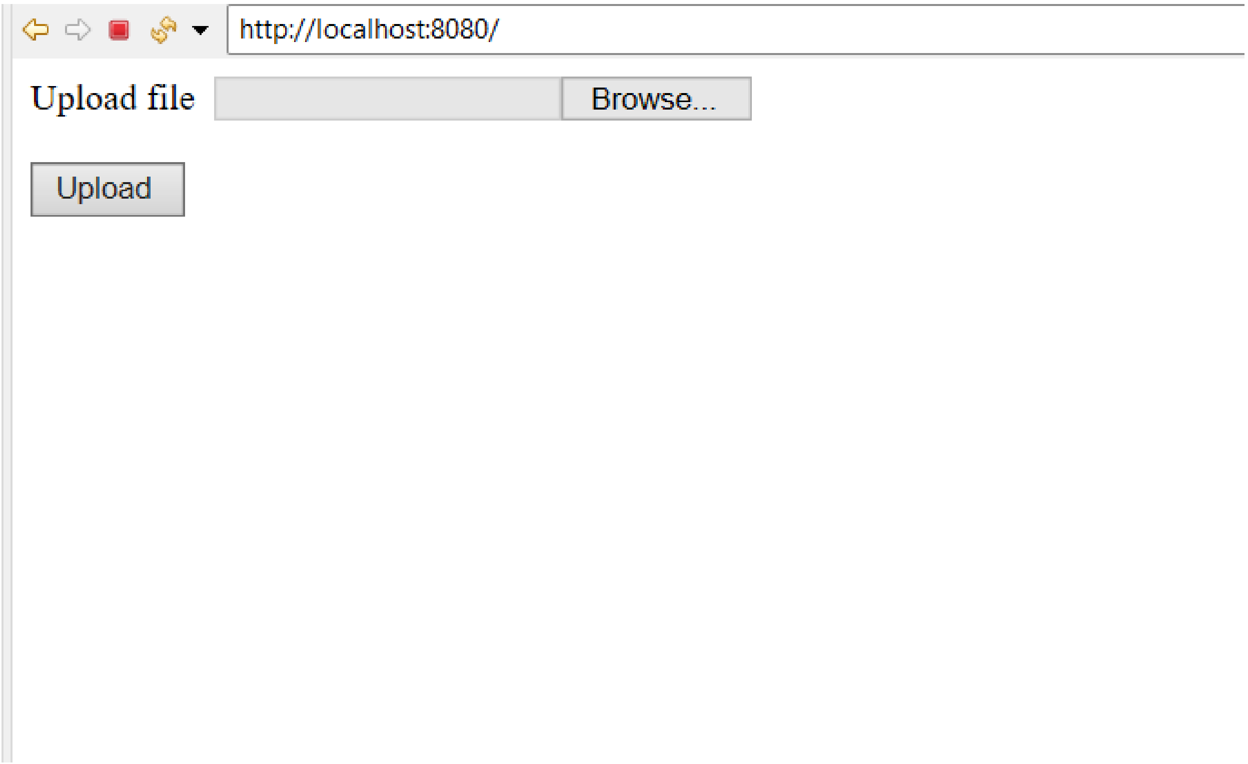
Problem Statement-3

```
package com.demo;
import org.springframework.boot.SpringApplication;
import org.springframework.cloud.netflix.zuul.EnableZuulProxy;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
@EnableZuulProxy
public class Springbootstarter1Application {
      public static void main(String[] args) {
            SpringApplication.run(Springbootstarter1Application.class, args);
      }
}
```

```
spring.application.name = zuulserver
zuul.routes.products.path = /api/demo/**
zuul.routes.products.url = http://localhost:8080/
server.port = 8111
```

Practice Question-4

```
@RequestMapping(value = "/")
       public String index() {
       return "index.html";
       }

       @SuppressWarnings("finally")
                  @RequestMapping(value = "/upload", method =
RequestMethod.POST, consumes = MediaType.MULTIPART_FORM_DATA_VALUE)
       public String fileUpload(@RequestParam("file") MultipartFile file) {
       String result = "File was uploaded successfully";

       try {
       File convertFile = new File("/var/tmp/"+file.getOriginalFilename());
       convertFile.createNewFile();
       FileOutputStream fout = new FileOutputStream(convertFile);
       fout.write(file.getBytes());
       fout.close();

       } catch (IOException iex) {
       result = "Error " + iex.getMessage();

       } finally {
       return result;
       }
       }

       @RequestMapping(value = "/download", method = RequestMethod.GET)
       public ResponseEntity<Object> downloadFile() throws IOException {
       String fileName = "static/dump.txt";
       ClassLoader classLoader = new
MainController().getClass().getClassLoader();

       File file = new File(classLoader.getResource(fileName).getFile());

       InputStreamResource resource = new InputStreamResource(new
FileInputStream(file));
       HttpHeaders headers = new HttpHeaders();

       headers.add("Content-Disposition", String.format("attachment;
filename=\"%s\"", file.getName()));
       headers.add("Cache-Control", "no-cache, no-store, must-revalidate");
       headers.add("Pragma", "no-cache");
       headers.add("Expires", "0");

       ResponseEntity<Object>
       responseEntity =
ResponseEntity.ok().headers(headers).contentLength(file.length()).contentType(
       MediaType.parseMediaType("application/txt")).body(resource);

       return responseEntity;
       }
```



Practice Question-5

```
            package com.demo.controller;
import java.util.Calendar;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.kafka.core.KafkaTemplate;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class MainController {

                    @Autowired
                    private KafkaTemplate<String, String> kafkaTemplate;

                    @RequestMapping(value = "/")
                   public String index() {
                    this.sendMessage("This is a message sent at " +
Calendar.getInstance().getTime());

                    return "Check Eclipse console for kafka output";
                   }
                    private void sendMessage(String msg) {
```

```
                    kafkaTemplate.send("ecommerce", msg);
                    }
```

Practice Question-6

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
@Controller
public class MainController {
@Autowired
private ProductRepository repository;
@RequestMapping("/")
@ResponseBody
public String index() {
return “This is running under SSL”;
}
```